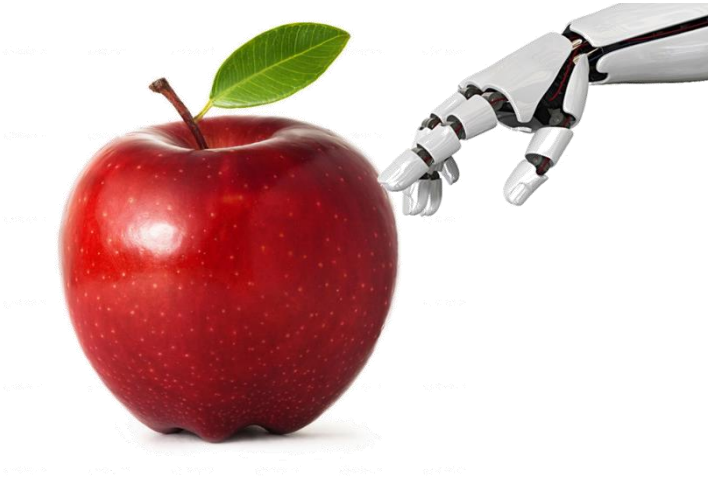
ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia Mecatrônica
Instrumentação
Profs. Larissa Driemeier/Marcilio Alves/Rafael T. Moura

Análise de Sinais usando MatLab



A aula de hoje será de estudo dirigido. Para isso você precisará do MATLAB® ou Octave.

Nas próximas páginas você encontrará nove exercícios, alguns resolvidos, outros parcialmente resolvidos e alguns você terá que encontrar a saída. Os exercícios têm o objetivo de fazer você entender um pouco mais sobre amostragem, leakage, windowing, filtros.

Não se preocupe em criar nenhum arquivo com explicações e respostas. **Somente as resoluções dos exercícios 3-4 e 7-9a devem ser disponibilizados no STOA até as 12hs00 de hoje. Coloque três arquivos ".m", zipados. Quaisquer explicações necessárias, inclusive os nomes dos integrantes da dupla, devem estar nos comentários dos arquivos.** Todos os gráficos devem ter título, título dos eixos (com unidades) e legenda. A aula é também de estudo, faça suas anotações.

Ao analisar um exercício **com solução**, seja detalhista. Não deixe de entender o código. Isso será imprescindível para que você consiga fazer os exercícios **sem solução**.

A lista é longa e o tempo curto... Não se distraia com conversas paralelas.

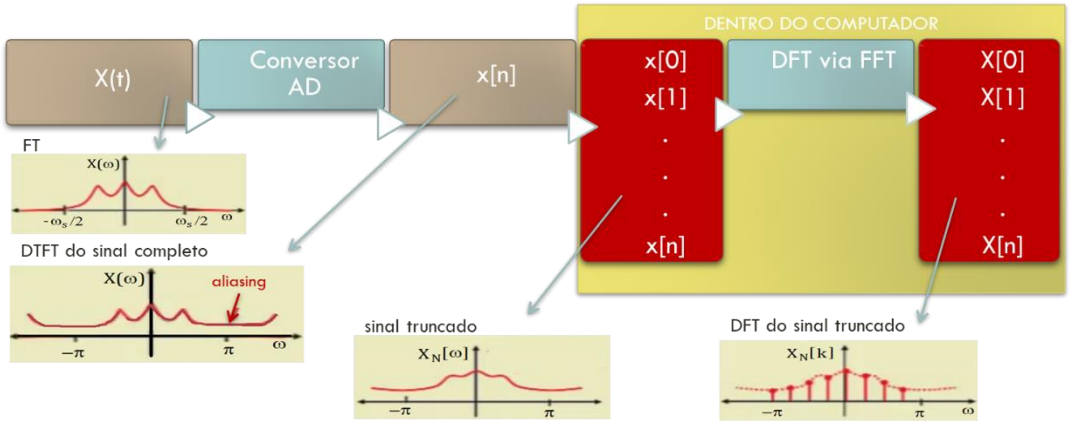
Boa aula!

Ferramentas matemáticas nos ensinam a fazer a FT (Transformada de Fourier) de um sinal contínuo e não periódico no tempo

A DTFT (Transformada de Fourier em Tempo Discreto) é a FT de um sinal amostrado em um intervalo de tempo infinito. Sua saída é periódica e contínua em frequência.

A DFT (Transformada de Fourier Discreta) pode ser vista como a versão de amostragem finita da DTFT. Ela é usada para calcular o espectro frequência de um sinal discreto no tempo usando o computador, já que os computadores só podem lidar com um número finito de valores. A DFT e a sua inversa estão implementadas no Matlab como fft and ifft.

Quais os erros que se comete entre a FT(o que queremos) e a DFT(o que medimos)? E como podemos evitar ou dimuniir esses erros?



Nomenclatura da frequência

Frequência em rad/s, Hertz e radianos/amostra

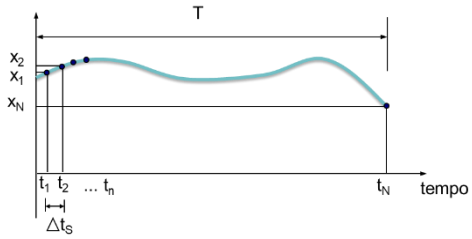
Importante ressaltar que, neste trabalho, segue-se a convenção adotada na maioria dos livros, i.é, ω é a frequência em rad/s, enquanto f é a frequência em $Hertz = ciclos/segundo$. São relações importantes, que muitos alunos confundem,

$$T = \frac{2\pi}{\omega}, \omega = 2\pi f$$

onde T é o período, em $segundos$.

Um sinal analógico $x(t)$ é definido por parâmetros de frequência e tempo. Supõe-se que o sinal seja amostrado por um período T, a uma freqüência de amostragem f_s,

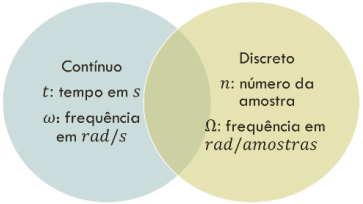
$$f_s = \frac{1}{\Delta t_s} = \frac{N}{T}$$



onde N é o número total de amostras. Para manter a analogia com sinais discretos, usa-se n (o número da amostra) como a unidade de tempo discreta. A relação entre n e Δt é,

$$t_n = n\Delta t_s$$

Chama-se **Frequência Digital** e usa-se, em geral, a letra grega Ω para representá-la,



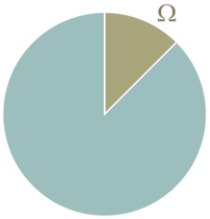
$$\Omega = 2\pi \frac{\omega}{\omega_s}\ rad/amostra$$

$$F = \frac{f}{f_s}\ ciclos/amostra$$

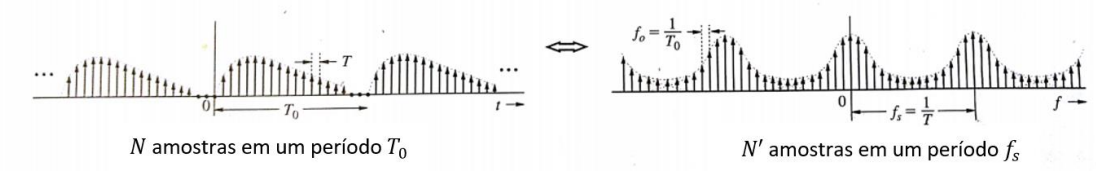
onde ω é a frequência do sinal e $\omega_s = 2\pi f_s$ a frequência de amostragem. A frequência, portanto, terá unidade de *radianos/amostra*, já que a unidade de tempo passou a ser *número da amostra*. A frequência de um sinal discreto é diferente da frequência tradicional de um sinal contínuo.

Por exemplo, se tivermos um sinal de freqüência de $10\ Hz$ e amostrarmos com uma freqüência de amostragem $f_s = 80\ Hz$, então sua freqüência digital é

$$\Omega = \frac{2\pi \times 10}{80} = \frac{2\pi}{8}$$



O que este número significa??? Isso significa que cada amostra move o sinal em Ω radianos. Se um ciclo contém 2π radianos, e cada amostra cobre $\frac{2\pi}{8}$rad, então, 8 amostras são necessárias para completar um ciclo.



N amostras em um período T_0 $\quad\Leftrightarrow\quad$ N' amostras em um período f_s

Pela Figura, adaptada de [3], o número N de amostras do sinal em um período T_0 é idêntico ao número de amostras do espectro, N', em um período f_s. A razão é,

$$T_0 = NT, \qquad \therefore N = \frac{T_0}{T}$$

$$f_s = \frac{1}{T} \qquad f_0 = \frac{f_s}{N} = \frac{1}{TN} = \frac{1}{T_0}$$

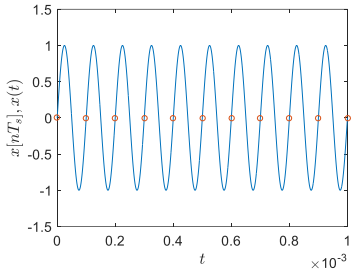
f_0 tambem e chamada de f_{bin}, i.e, a resolução do domínio da frequência.

Exercício 01

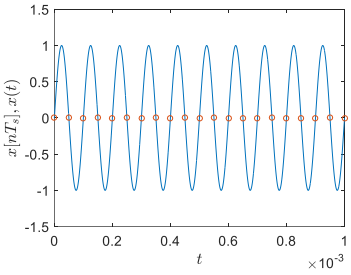
Vamos produzir uma senoide no MatLab e amostrá-la. Explore especialmente a influência da taxa de amostragem no resultado do sinal amostrado (vetor `sinal_sample`), comparando-o com o sinal contínuo (vetor `sinal`).

```
clear all; close all; clc
%% dados do sinal
f = 10000;%Freq entrada Hz
fs =20000;% Frequencia de amostragem Hz
%% gerar sinal 'contínuo' com 10 períodos. Para isso:
% alta discretização (taxa de 100*f)
% T=10, ié, 10 períodos do sinal
tempo = [0:1/(100*f):10/f];
sinal = sin(2*pi*f*tempo); % Geração onda senoidal em uma
%% plotar sinal
plot(tempo,sinal)
hold;
%% sinal amostrado
Ts = 1/fs;
N=length(tempo);
n = [0:1:N-1];
t_sample = [0 : Ts : n(N)*Ts];
DigitalFrequency=2*pi*f/fs;
sinal_sample = sin (DigitalFrequency.*n);
plot(t_sample, sinal_sample,'o');
axis([0 10/f -1.5 1.5])
set(gca,'FontSize',16)
set(gca,'FontSize',16)
xlabel('$t$','Interpreter','LaTex','FontSize',18)
ylabel('$x[nT_s],x(t)$','Interpreter','LaTex','FontSize',18)
```

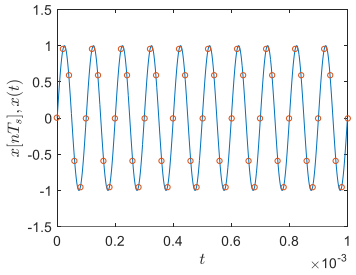
Nessas linhas você define o número de pontos amostrados `N`, período de amostragem `Ts` e, como consequência, o intervalo de tempo do sinal amostrado `(N-1)*Ts`. Mantendo `N` constante e aumentando `fs` você diminui o intervalo e vice versa.



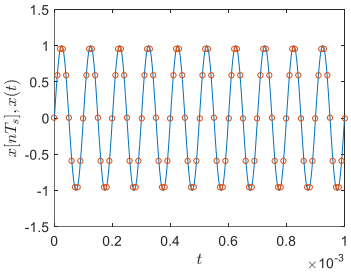
Sinal de 10 kHz com taxa de 10kHz



Sinal de 10 kHz com taxa de 20kHz



Sinal de 10 kHz com taxa de 50kHz



Sinal de 10 kHz com taxa de 100kHz

Exercício 02

FFT (FAST FOURIER TRANSFORM) é simplesmente uma forma mais rápida de calcular a DFT: A FFT utiliza alguns algoritmos que permitem reduzir o número de operações para $N\log_2 N$. Para utilizar a FFT, é necessário que o número de amostras seja uma potência de 2 – a FFT é executada mais rapidamente com um vetor cujo comprimento é uma potência de 2.

Para $N = 1000$, $DFT = 1\,000\,000$, $FFT = 10\,000$ operações

A FFT no Matlab: Matlab permite o cálculo fácil da DFT via FFT. Se tivermos um vetor A, de n elementos,

$$>> FFTdeA = fft(A);$$

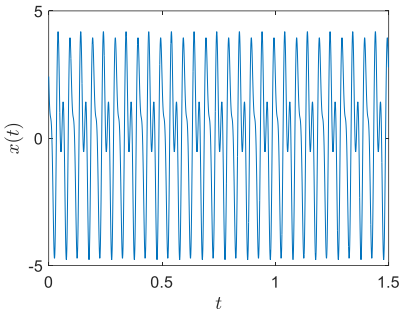
Agora vamos utilizar a FFT. No primeiro exemplo, aprenderemos a analisar a resposta da FFT. No segundo, apenas iremos ilustrar o papel da função `fftshift`. Por último, utilizaremos algumas propriedades bastante conhecidas e importantes da transformada. Como exemplo, considera-se uma função cosseno,

$$x(t) = 3\cos(2\pi f_1 t + 0.2) + \cos(2\pi f_2 t - 0.3) + 2\cos(2\pi f_3 t + 2.4)$$

Considere a frequência de amostragem $f_s = 1000$ amostras/segundo, amostrando-se em um período total de 1.5 ms.

Com esse exemplo, vamos entender a FFT...

```
clear all; close all; clc

%% Senoides e aliasing
% Taxa de amostragem
fs=1000; Ts=1/fs;
% Frequencias, em Hz, do sinal
f1=20;   DigFreq1=2*pi*f1/fs;
f2=30;   DigFreq2=2*pi*f2/fs;
f3=40;   DigFreq3=2*pi*f3/fs;
%Plot
N=1500;
n = 0:1:N-1;
t_sample = [0 : Ts : (N-1)*Ts];
x=3.*cos(DigFreq1.*n+0.2)+cos(DigFreq2.*n-0.3)+2.*cos(DigFreq3.*n+2.4);
figure;
plot(t_sample,x)
set(gca,'FontSize',16)
set(gca,'FontSize',16)
xlabel('$t$','Interpreter','LaTex','FontSize',18)
ylabel('$x(t)$','Interpreter','LaTex','FontSize',18)
```




```
%% FFT do sinal
X=fft(x);
```

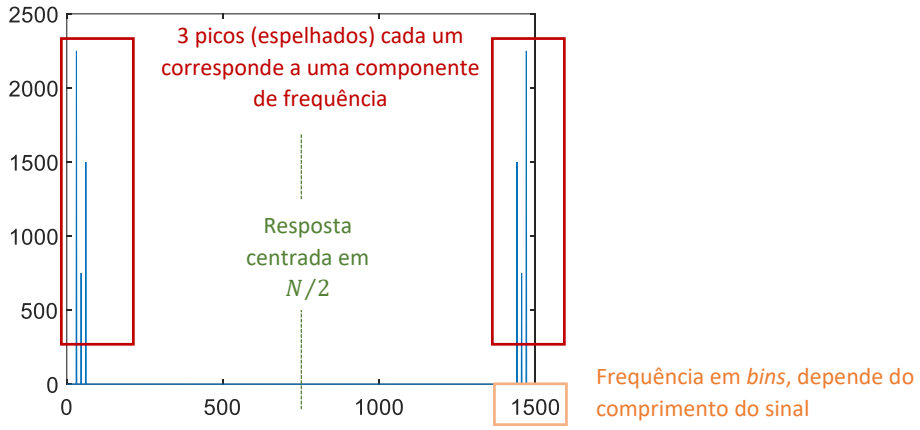
O comprimento das variáveis `x` (domínio do tempo) e `X` (domínio da frequência) são iguais. Verifique!

```
% os valores de x são valores reais, enquanto os valores de X são
% complexos. Veja alguns exemplos:
x(2:6)
X(30:34)
```

```
ans =

    2.0717    1.7535    1.4794    1.2572    1.0884


ans =

   1.0e+03 *

  -0.0000 - 0.0000i   2.2051 + 0.4470i   0.0000 - 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i
```

```
%X são valores complexos porque representam magnitude e fase!!!!
%Magnitude
figure;
X_mag=abs(X);
X_mag(30:34)
plot(X_mag)
set(gca,'FontSize',16)
```

```
ans =

   1.0e+03 *

    0.0000    2.2500    0.0000    0.0000    0.0000
```



A faixa de frequência de um gráfico de espectro depende da taxa de amostragem. I.é, faz sentido analisar somente até frequências que sejam metade da frequência de aquisição do sinal (lembre-se da frequência de Nyquist!!!).

Porque os valores espelhados? Sabe-se que a transformada de Fourier de um sinal discreto é um sinal periódico no domínio da frequência. O Matlab, obviamente, sempre trabalha em domínio discreto, e, portanto, usando a função `fft`, você sempre obtém sinais de frequência periódicos (veja a figura extraída de [3] que usamos para definir frequência digital na pag.2).

Portanto, faz sentido que o MatLab considere apenas um período desse sinal (portanto, apenas as primeiras `N/2` amostras espelhadas – manter o espelho é importante matematicamente para a inversa). Como a primeira amostra vetorial $x(t)$ corresponde à amostra no tempo $t = 0$, então a primeira amostra de sinal $X(f)$ corresponde à amostra em $f = 0$ (ié, sempre representa DC). Porém, o MatLab não permite que uma *array* tenha índices negativos ou nulos. Por esta razão, após uma `fft`, o sinal obtido é aquele na Figura, onde as amostras de frequência negativa são rebatidas ao fundo, na segunda metade do vetor.

No eixo das abscissas, correspondente a frequências, os valores de X variam 0-1499, que é o tamanho do sinal criado por nós. Essas são chamadas de *bin frequencies*. A resolução em frequência depende da relação entre a taxa de amostragem do sinal de entrada (f_s) e o comprimento da FFT (N):

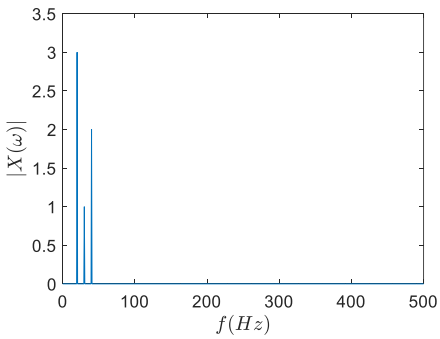
$$\Delta f_{bins} = \frac{f_s}{N}$$

de modo que frequência em *bins* pode ser facilmente convertida em frequência em Hz,

$$f_n = n\Delta f_{bins} = n\frac{f_s}{N}$$

No eixo das ordenadas, a magnitude está multiplicada também por $N/2$.

```
figure;
n=0:1:N-1
plot((fs/N).*n,X_mag/(N/2))
xlim([0 fs/2])
set(gca,'FontSize',16)
xlabel('$f(Hz)$','Interpreter','LaTex','FontSize',18)
ylabel('$|X(\omega)|$','Interpreter','LaTex','FontSize',18)
```



A fase do sinal também é uma informação importante, que pode ser tirada de X.

```
%Fase
X_fase=angle(X);
%se você verificar a fase em cada componente verá que coincide com a
% fase do sinal
%
X_fase(31)
X_fase(46)
X_fase(61)
```

De onde saíram esses números, 31, 46 e 61? ☺

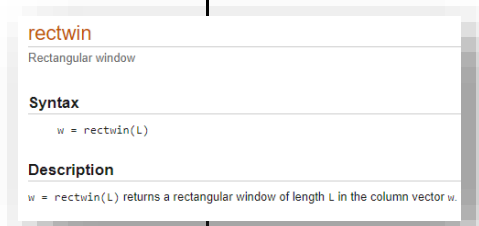
```
ans =

    0.2000


ans =

   -0.3000


ans =

    2.4000
```

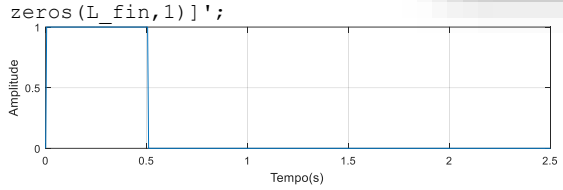
Finalmente, plote o gráfico com a magnitude em decibeis, usando o seguinte comando,

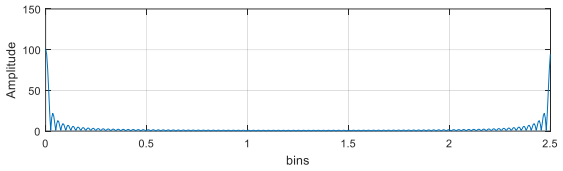
`mag2db(X_mag/(N/2))`, i.e., `10*log10(X_mag/(N/2))`

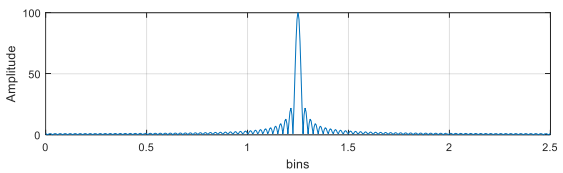
Apenas para ilustrar o comando `fftshift`, verifique as imagens abaixo,

```
clear all; close all; clc
%%
Tmax=0.5;                % Intervalo de duração de cada onda
fs=200;                  % Frequência de amostragem
t=[0:(1/fs):Tmax+2];     % Amostragem no tempo
L=length(t);

% Pulso retangular
T0=0;                    % Instante de início do pulso retangular
T=Tmax;                  % Duração do pulso retangular

% Definição do início e final da janela
L_ini=length([0:(1/fs):T0]);
L_pulse=length([0:(1/fs):T]);
L_fin=L-L_ini-L_pulse;

win = rectwin(L_pulse);
wRect1 = [zeros(L_ini,1); win; zeros(L_fin,1)]';
figure(1)
subplot(311)
plot(t,wRect1,'LineWidth',1)
grid on
xlabel('Tempo(s)');
ylabel('Amplitude');


X=fft(wRect1);
subplot(312)
plot(t,abs(X))
grid on
xlabel('bins');
ylabel('Amplitude');

Y=fftshift(X);
subplot(313)
plot(t,abs(Y))
ylim([0 100])
grid on
xlabel('bins');
ylabel('Amplitude');
```



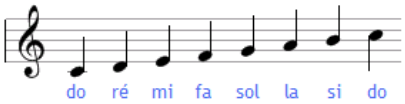






Portanto, a função `shiftfft` permite que você reordene as frequências e represente o sinal transformado, centralizado no vetor de frequência, como esperado matematicamente. Plot `ifft` (inversa da fft) de ambos os sinais – com e sem o `shiftfft` e comente os resultados.

Exercício 3



Os arquivos '.wav' na pasta *NotasMusicais* deveriam estar numeradas em ordem crescente de frequência, que coincide com as notas musicais **Dó(261.63 Hz), Ré(293.66 Hz), Mi(329.63 Hz), Fá(349.23 Hz), Sol (392 Hz), Lá (440 Hz), Si (493.88 Hz)**. Alguém não fez o trabalho de forma adequada. Sua tarefa consiste em criar um programa Matlab® que reconheça a sequência de notas, de forma a completar a tabela:

Nota musical	Nome do arquivo
Dó	
Ré	
Mi	
Fá	
Sol	
Lá	
Si	

Use o comando `audioread`, do MATLAB.

Exercício 4

O sinal de frequência 1Hz

$$x = \cos(2\pi t)$$

é amostrado a uma taxa de 10Hz, pelo período total de 3s (isto é, foram amostrados 3 períodos do sinal). Serão analisadas três situações:

a. **Truque do Zero padding**:

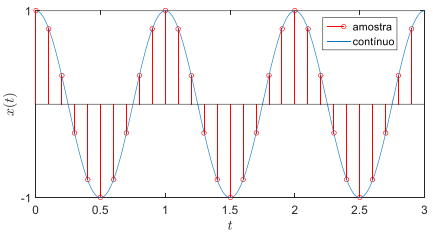
Depois de coletados os N pontos de amostragem, colocamos alguns zeros adicionais ao final da lista para *enganar* o processo (como são zeros não alteram os valores da Transformada de Fourier Discreta).

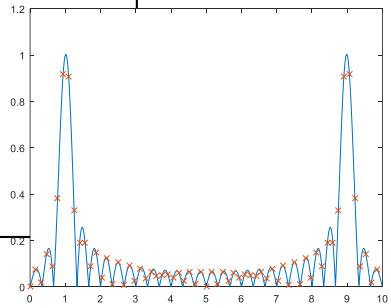
No MatLab, $X = fft(x, N_p)$

Use os valores $N_p = 30, 64, 128, 256$ **mantendo a frequência de amostragem**. Avalie o resultado comparando as respostas. Para comparar as respostas, considere a resposta com $N_p = 1024$ como contínua e faça quatro gráficos, comparando a resposta contínua com aquelas com diferentes valores de N_p . Abaixo, como ilustração, o gráfico de comparação para $N_p = 64$. Responda:

i. qual a influência de acrescentar zeros no final do sinal amostrado (*truque do zero-padding*)?
ii. Para um sinal com aliasing, que leve a um espectro errado no domínio da frequência, qual a influência de usar *truque do zero-padding*?

```
clear all; close all; clc

%% Senoide e fft - zero padding
N=30;
fs=10;
n = [0:N-1];
t=0:0.01:N/fs;
x_cont= cos(2*pi*t);
x = cos(2*pi*n/10);

N2 = 64;
n2=0:1:N2-1;
N5=2048;
n5=0:1:N5-1;
 %%%
 X2 = abs(fft(x,N2))./(length(x)/2);
 X5 = abs(fft(x,N5))./(length(x)/2);
figure;
plot((fs/2)/(N5/2).*n5,X5)
hold on
plot((fs/2)/(N2/2).*n2,X2,'x')
```

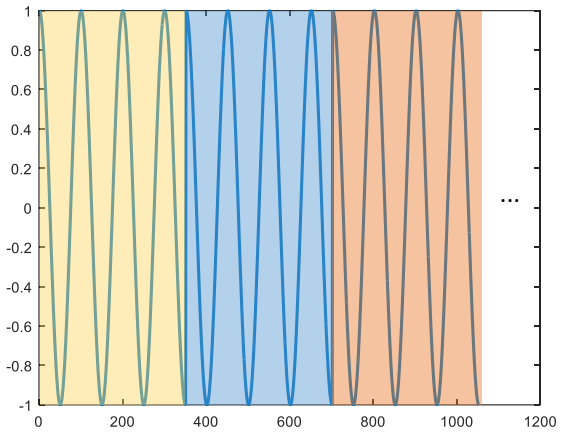




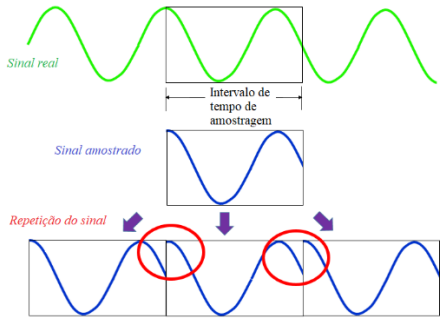
b. **Número de períodos de amostragem**: Mantendo o valor de $N_p = 1024$ constante, plote a `fft` para diferentes números de períodos amostrados. Você amostrou 3 períodos no item (a). Compare essa resposta com a resposta obtida amostrando 6, 9 e 12 períodos. Existe uma maneira rápida de aumentar o número de períodos no MatLab,

```
%% Aumentando o número de períodos
x1 = cos(2*pi*n/10); % 3 periodos
x2 = [x1 x1]; % 6 periodos
x3 = [x1 x1 x1]; % 9 periodos
x4 = [x1 x1 x1 x1]; % 12 periodos
```

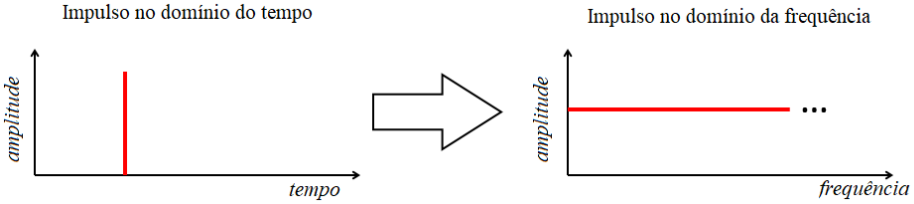
c. Se a amostragem do sinal não cobrir ciclos inteiros, o espectro calculado irá apresentar o fenômeno de vazamento espectral ("leakage"). Veja na figura abaixo a ilustração de como o sinal "é visto" pela transformada. Discuta o fenômeno de *leakage*, definindo períodos não completos de amostragem do exemplo.




```
%% Senoide e fft - leakage
N=35;
fs=10;
n = [0:N-1];
t=0:0.01:N/fs;
x_cont= cos(2*pi*t);
x = cos(2*pi*n/10);
%% Aumentando o número de períodos
x1 = cos(2*pi*n/10); % 3 periodos
x2 = [x1 x1]; % 6 periodos
x3 = [x1 x1 x1]; % 9 periodos
x4 = [x1 x1 x1 x1]; % 12 periodos
x_cont3=[x_cont x_cont x_cont];
figure;
plot(x_cont3,'Linewidth',2);
```

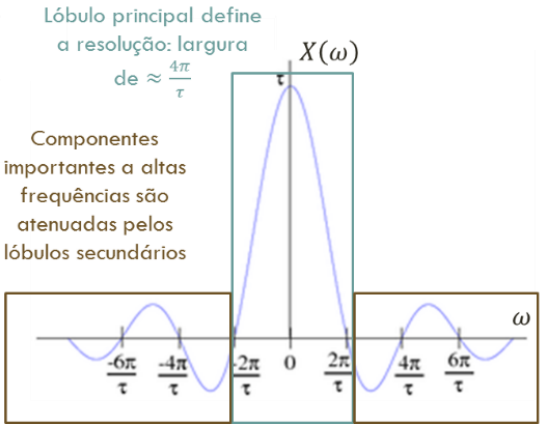


Existem transições repentinas no final de cada sinal capturado. Esses transientes agudos têm uma ampla resposta de frequência. Sinais transientes curtos no domínio do tempo produzem frequência de banda larga, como mostrado na Figura abaixo. Como consequência a um *espalhamento da energia*, i.é, as amplitudes calculadas sofrem um achatamento e um espalhamento em torno das valores espectrais originais.



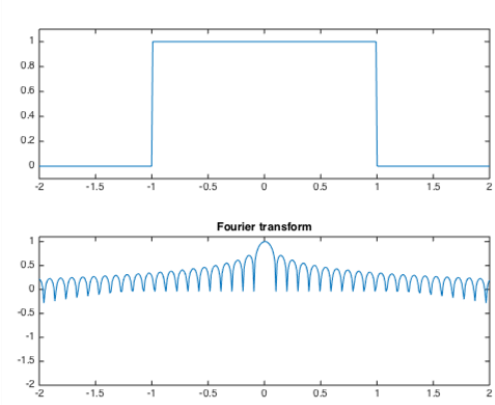
Exercício 5

Uma solução para o *leakage* é o janelamento (windowing). Diferentes tipos de janelas podem ser utilizados. A mais simples é a retangular, que é igual a 1 durante o intervalo de tempo que se pretende analisar, e igual a zero fora desse intervalo. Foi essa janela que usamos no exercício 2. Já aprendemos que a janela retangular no domínio do tempo, resulta em uma função *sinc* no domínio da frequência (figura ao lado).



Quanto mais estreito for o lóbulo principal, melhor a resolução frequencial. No entanto, quanto mais estreito o lóbulo principal, mais altos se tornam os lóbulos laterais.

E, consequentemente, uma janela retangular que fornece boa resolução frequencial, possui lóbulos laterais muito altos, resultando em muito ruído de fundo. Além disso, ocorre o fenômeno de *leakage*, discutido anteriormente, se os períodos de amostragem não forem completos.



Existem várias janelas, além da retangular, cujo objetivo é diminuir a influência dos extremos da amostragem. Veja, por exemplo, a janela de Hanning (também chamada de Hann), em homenagem ao vienense Julius Ferdinand von Hann (1839-1921). A janela Hanning é modelada como:

$$w(k) = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi k}{M-1}\right)\right] \qquad k = 0, \dots, M-1$$

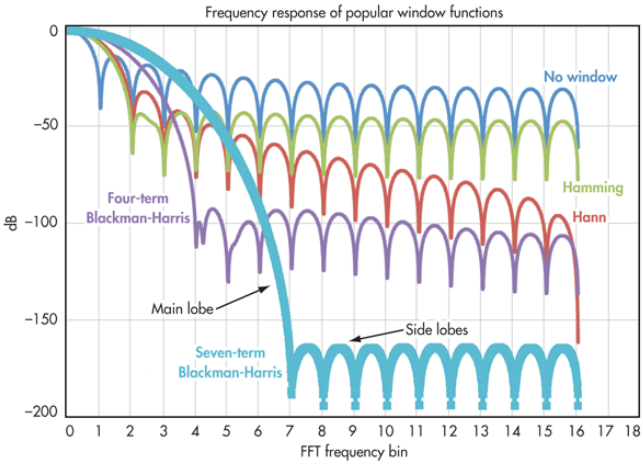
No MatLab, $w = hanning(M);$ ou $w = hann(M);$

O janelamento Hamming começa em 0,08, sobe para 1 no meio do período, e depois cai novamente até 0,08 no final.

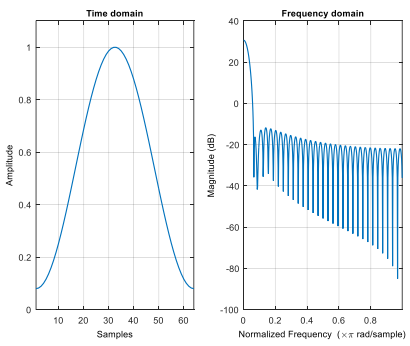
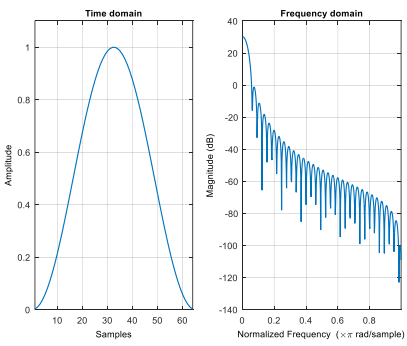
$$w(k) = 0{,}54 - 0{,}46\cos\left(\frac{2\pi k}{M-1}\right), \quad k = 0, \dots, M-1$$

Ou seja, os valores iniciais e finais da amostragem são atenuados. No MatLab, $w = hamming(M);$

A figura ao lado ilustra o comportamento de algumas janelas conhecidas. Verifique que -150dB correspondem a, aproximadamente, uma magnitude de 3.16e-8. *Um número maior que uma unidade apresenta um valor positivo em dB, enquanto um número menor que a unidade apresenta valor negativo.*



Use os seguintes comandos para entender as janelas Hamming e Hanning,

`L = 64;` `wvtool(hamming(L))`	`L = 64;` `wvtool(hanning(L))`
	

Veja que a ferramenta `wvtool` (WindowVisualization Tool), abre uma janela com a plotagem no tempo e na frequência do sinal. Use `wvtool` e compare as janelas Hamming, Hann, e Gaussiana

```
wvtool(hamming(64),hann(64),gausswin(64))
```

Existem outras janelas (vide Kaiser em https://ccrma.stanford.edu/~jos/sasp/Kaiser_Window.html ou https://www.mathworks.com/help/signal/ref/kaiser.html). Nessa aula, usaremos a Hamming, que é mais simples e, portanto, a mais usada. A janela Hanning tem desempenho satisfatório em, aproximadamente, 95% dos casos. Tem boa resolução em frequência e reduzido *leakage* spectral.

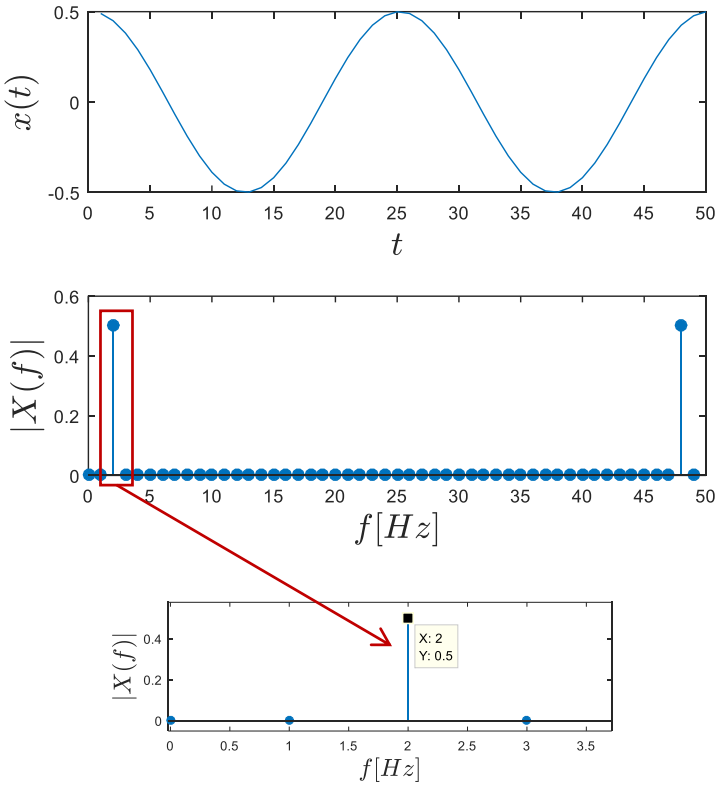
Exercício 6

Considere o sinal,

$$x = 0.5\cos(2\pi f_1 n + 0.2)$$

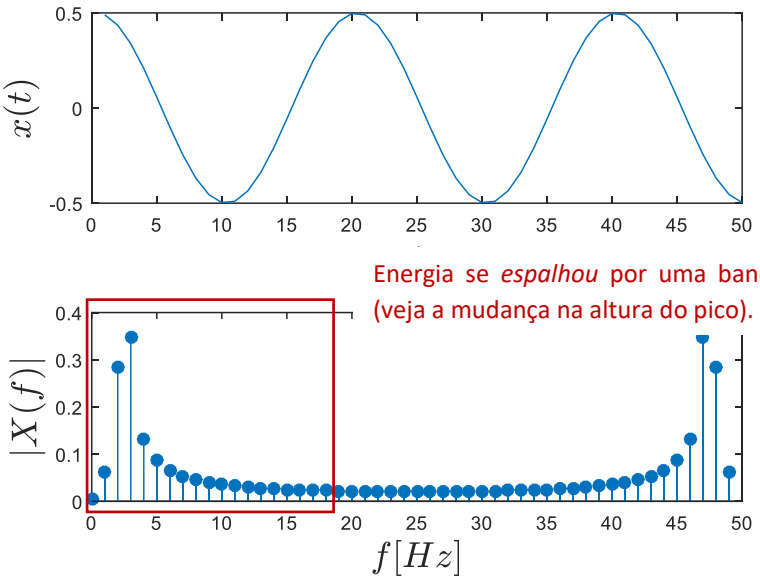
Segundo o código MATLAB abaixo,

```
%% sinal sem leakage
fs=50;
t=0:1/fs:1-1/fs;
f1=2;
x1=0.5*cos(2*pi*f1*t+0.2);
subplot(2,1,1)
plot(x1)
xlabel('$t$','Interpreter','LaTex','FontSize',18)
ylabel('$x(t)$','Interpreter','LaTex','FontSize',18)
%
X1=fft(x1);
subplot(2,1,2)
N=length(X1);
N1=length(X1);
stem([0:length(X1)-1]*fs/N,abs(X1)/(N/2),'filled')
xlabel('$f [Hz]$','Interpreter','LaTex','FontSize',18)
ylabel('$| X(f)|$','Interpreter','LaTex','FontSize',18)
```



Se mudarmos a frequência de 2 para 2.5Hz, a amostragem não será de período inteiro e a resposta muda radicalmente.

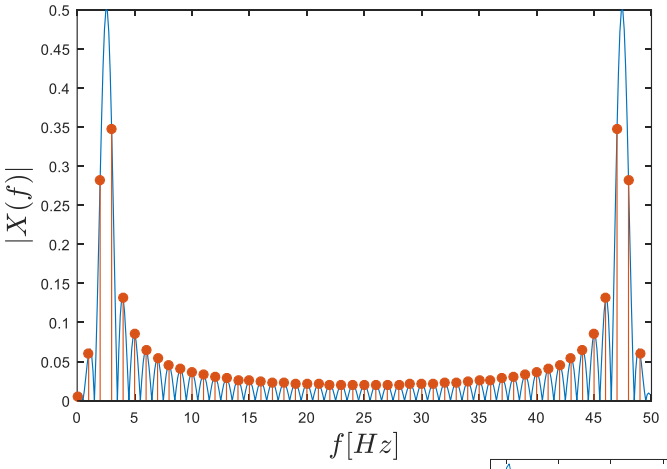
```
%% sinal com leakage
figure;
fs=50;
t=0:1/fs:1-1/fs;
f1=2.5;
x1=0.5*cos(2*pi*f1*t+0.2);
subplot(2,1,1)
plot(x1)
xlabel('$t$','Interpreter','LaTex','FontSize',18)
ylabel('$x(t)$','Interpreter','LaTex','FontSize',18)
%
X1=fft(x1);
N=length(X1);
subplot(2,1,2)
stem([0:length(X1)-1]*fs/N,abs(X1)/(N/2),'filled')
xlabel('$f [Hz]$','Interpreter','LaTex','FontSize',18)
ylabel('$| X(f)|$','Interpreter','LaTex','FontSize',18)
```



Energia se *espalhou* por uma banda do espectro (veja a mudança na altura do pico).

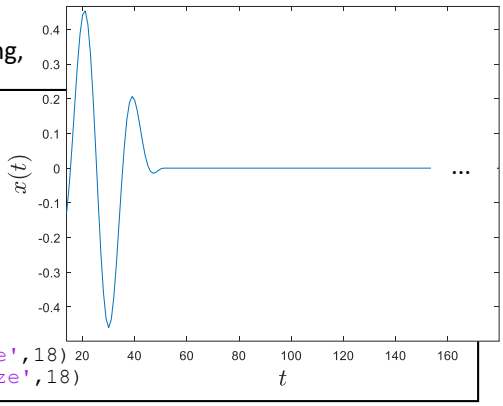
O uso do truque *zero padding* não **irá** ajudar. Veja que foi adicionado um número de zeros ao final do sinal (sem usar a variável N_p da fft) para termos controle do sinal. A resposta mostra o sinal bem mais discretizado, como previámos, mas espalhado em seu espectro (ou seja, ainda com leakage).

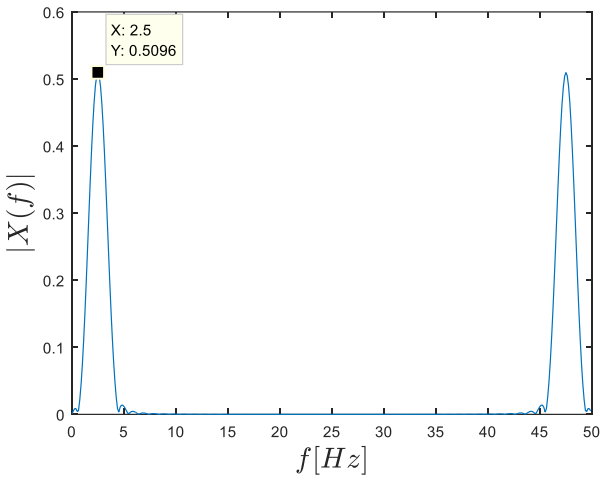
```
%% Zero-padding
figure;
x2=[x1 zeros(1,7*N)];
X2=fft(x2);
N2=length(X2);
plot([0:N2-1]*fs/N2,abs(X2)/((N2-7*N)/2))
hold on
stem([0:length(X1)-1]*fs/N,abs(X1)/(N/2),'filled')
xlabel('$f [Hz]$','Interpreter','LaTex','FontSize',18)
ylabel('$| X(f)|$','Interpreter','LaTex','FontSize',18)
```



Se usarmos o janelamento, **antes** do truque do zero-padding,

```
%% Zero-padding e windowing
figure;
L=length(x1);
x3=x1.*hanning(L)';
x3=[x3 zeros(1,7*N)];
X3=fft(x3);
N3=length(X3);
plot([0:N3-1]*fs/N3,abs(X3)/((N3-7*N)/4))
figure;
plot(x3)
xlabel('$f [Hz]$','Interpreter','LaTex','FontSize',18)
ylabel('$| X(f)|$','Interpreter','LaTex','FontSize',18)
```





O janelamento do sinal original altera a amplitude dos pontos de um sinal, o que resulta na perda da energia total. Para corrigir a distorção de amplitude ou energia, cada linha espectral de um espectro de frequência com janelamento é multiplicada por um fator fixo. Esse fator é determinado pelo tipo de janela que foi aplicado (veja tabela abaixo). Somente a janela Uniforme, que é equivalente a nenhuma janela, possui os mesmos fatores de correção de amplitude e energia.

Tipo de Janela	Correção de amplitude	Correção de energia
Uniforme	1.00	1.00
Hanning	2.00	1.63
Flattop	4.18	2.26
Blackman	2.80	1.97
Hamming	1.85	1.59
Kaiser-Bessel	2.49	1.86

Para janela Hanning, portanto, a regra é que a magnitude é o valor dividido por $N/4$, e a fase é o valor encontrado. A frequência segue a regra já definida:

```
angle(X3(21))
abs(X3(21))*4/(N3-Lzp)
20*fs/N3

ans =

    0.2001


ans =

    0.5096


ans =

    2.5000
```

Exercício 7

Analise o sinal sinusoidal composto de três frequências,

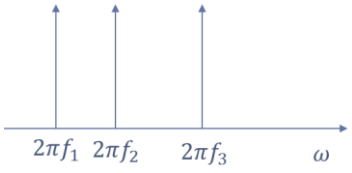
$$x = \cos(2\pi f_1 n) + \cos(2\pi f_2 n T_s) + \cos(2\pi f_3 n T_s)$$

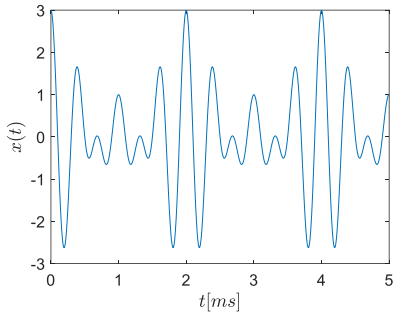
Onde,

$f_1 = 2000$ Hz

$f_2 = 2500$ Hz

$f_3 = 3000$ Hz





$f_s = 1000$ Hz, onde f_s é a taxa de amostragem, e $T_s = \frac{1}{f_s}$ é o período de amostragem.

Nesse exercício você deve definir uma frequência de amostragem constante, f_s, e utilizar um número diferente de amostras: $N = 10, 20, 40\ e\ 100$.

Utilize sempre as janelas retangular e hamming, conforme parte do código abaixo.

Analise a resposta das duas janelas a medida em que o período de amostragem (isto é, N) aumenta.

Discuta os resultados.

```
fs=10000; T=1/fs;  % Frequencia e período de amostragem
%Frequencias do sinal
f1=2e3;
f2=2.5e3;
f3=3e3;
% Frequencias para cálculo da DFT
w = 0:pi/1024:pi-pi/1024; %comprimento de w é usado em zero padding
figure(1)

L = 10;             % Define Numero de amostras
n=0:L-1;
x=cos(2*pi*f1*T.*n)+cos(2*pi*f2*T.*n)+cos(2*pi*f3*T.*n); % Calcular x(n)*w(n)
X=fft(x,length(w));
subplot(221)
plot(w./pi,abs(X))
axis([0 1 0 50])
title('Rectangular Window -- L = 10')

 h = hamming(L);       % Hamming window
 xh=x.*h';
 Xh=fft(xh,length(w));
 subplot(222)
 plot(w./pi,abs(Xh))
 axis([0 1 0 50])
 title('Hamming Window -- L = 10')
```

Exercício 8

Para detalhar outro assunto importante, vamos testar alguns filtros.

Inicialmente, vamos fazer um exemplo filtro de média móvel (*moving average filter).* É feita uma média de um número `M` de pontos do sinal da entrada `x[i]`, para produzir cada ponto do sinal de saída `y[i]`:

$$y[i] = \frac{1}{M}\sum_{j=1}^{m} x[i+j]$$

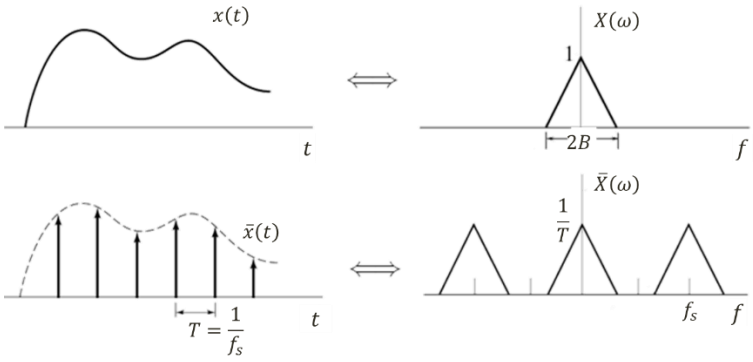
```
clear all; close all; clc
t = linspace(-pi,pi,100);
x = sin(t) + 0.25*rand(size(t));
%
windowSize = 5;
b = (1/windowSize)*ones(1,windowSize);
a = 1;
y = filter(b,a,x);
%
plot(t,x)
hold on
plot(t,y)
legend('Dados com ruído','Dados filtrados')
```

Veja o script acima. Foi usado o comando filter do MatLab, para simular um filtro de média móvel, `filter(a,b,x)`

$$\begin{aligned} a(1)y(n) \;=\; & b(1)x(n) + b(2)x(n-1) + \ldots + b(N_b)x(n-N_b+1) \\ & - a(2)y(n-1) - \ldots - a(N_a)y(n-N_a+1) \end{aligned}$$

Atenção: verifique o uso do comando `filter`.

Outro ponto importante para se tratar de filtro, é o uso de filtro anti aliasing. A figura abaixo ilustra a diferença entre a Transformada de Fourier de um sinal discreto e de um sinal contínuo. Isto é, quando o sinal é amostrado, seu espectro é replicado de acordo com a frequência de amostragem f_s.



$$\bar{x}(t) = \sum_k x(t)\,\delta(t-kT) \qquad \bar{X}(\omega) = \frac{1}{T}X(\omega - n\omega_s)$$

Por isso, o diz o teorema de Nyquist que:

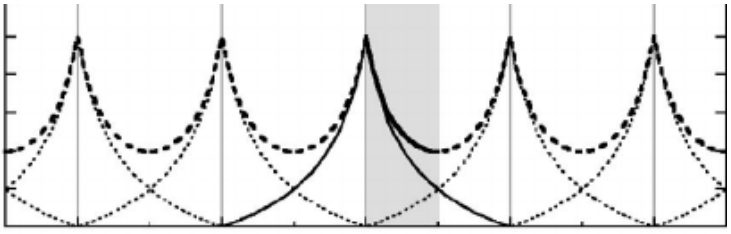
Se um sinal analógico $x(t)$ tem banda limitada, ou seja, se a frequência mais elevada do sinal é B ou seja,

$X(\omega) = 0$ para $|f| > B$,

então, é suficiente uma amostragem a qualquer taxa

$$f_s > 2B$$

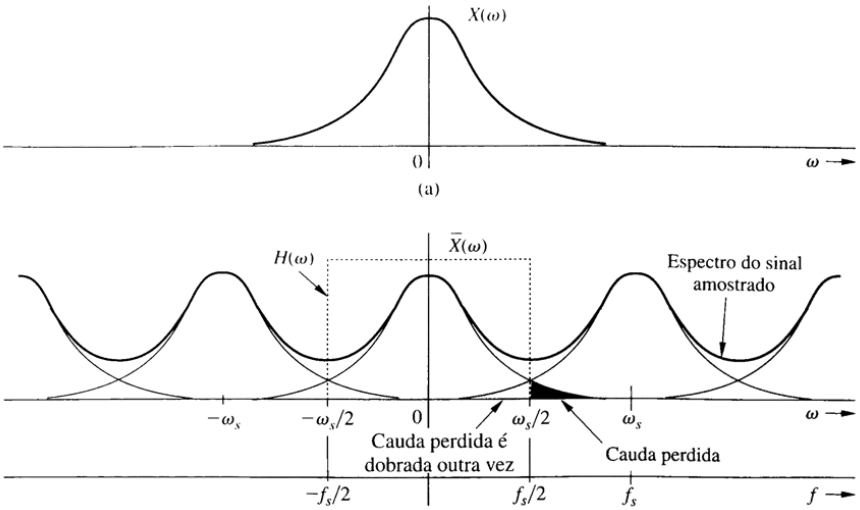
Porém, na maioria das vezes, não sabemos a priori, quais as frequências de nosso sinal. Além disso, muitos sinais não tem largura de banda finita ou não conhecemos. Dessa forma, existe uma grande chance de ocorrer sobreposição nas réplicas da frequência... o que impossibilitaria de reconstruir o sinal corretamente.



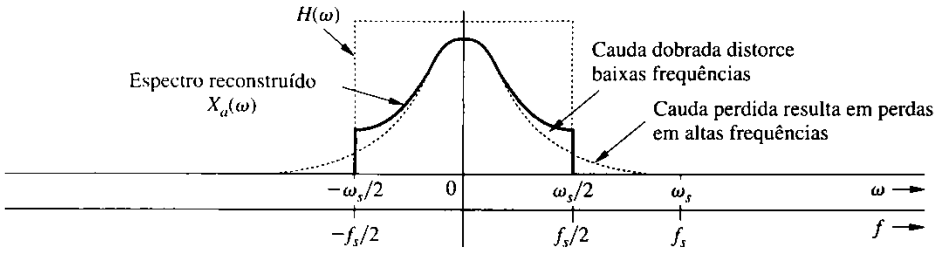
Quando há sobreposição de réplicas, diz-se que ocorreu *aliasing.*

Portanto, as frequências altas devem ser eliminadas **ANTES** da amostragem do sinal.

Como??? Emprega-se um filtro *passa-baixa* com frequência de corte $f_S/2$. Esse filtro é chamado de filtro *anti-aliasing.* O espectro das componentes de baixa frequência permanece intacto.

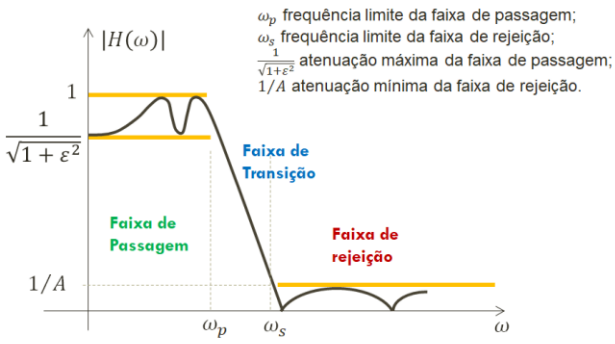


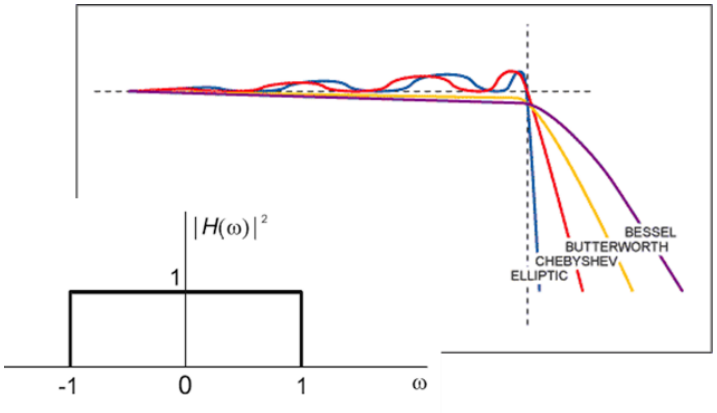
Como perdemos as componentes de alta frequência, determina-se a frequência de corte com base nas frequências de interesse do sinal e na frequência de amostragem. Para extrair corretamente a informação fundamental do sinal analisado é necessário selecionar as frequências de interesse que compõe esse sinal.



Como fazer isso?

Com um **FILTRO**. Filtros são SLIT capazes de modificar as características dos sinais de entrada de tal modo que apenas uma parcela específica dos seus componentes de frequência chega à saída do filtro. A resposta em frequência do filtro é caracterizada por **uma faixa de passagem** e **uma faixa de rejeição**, separadas por uma **faixa de transição ou faixa de guarda.**





Filtro passa baixa Butterworth

Para o filtro Butterworth, define-se

$$H(\omega) = \frac{1}{\sqrt{1+\varepsilon\left(\frac{\omega}{\omega_c}\right)^{2N}}}$$

Função `Buttord`

```
[N,Wc]=buttord(Wp,Ws,Rp,Rs)
```

`Wp` frequência limite da banda de passagem (em rad/s);
`Ws` frequência limite da banda de rejeição (em rad/s);
`Rp` máxima atenuação da banda de passagem (em dB, em `Wp`);
`Rs` mínima atenuação na banda de rejeição (em dB, em `Ws`)
OBS: O comando `buttord` define `Wc` usando a especificação de banda de rejeição e, possivelmente, deixando uma margem de segurança para banda de passagem.

ou função `butter`

```
[a,b]=butter(n,Wn)
```

retorna os coeficientes da função de transferência de um filtro passa-baixa de ordem `n` do tipo Butterworth com frequência de corte normalizada `Wn`.

Utilize um filtro tipo Butter e verifique se o problema de aliasing diminui. Perceba que `Wn` é a frequência de corte normalizada pela frequência de Nyquist, $2\pi f_s/2$. Isso quer dizer que `Wn=1`, a frequência de corte é a frequência de Nyquist.

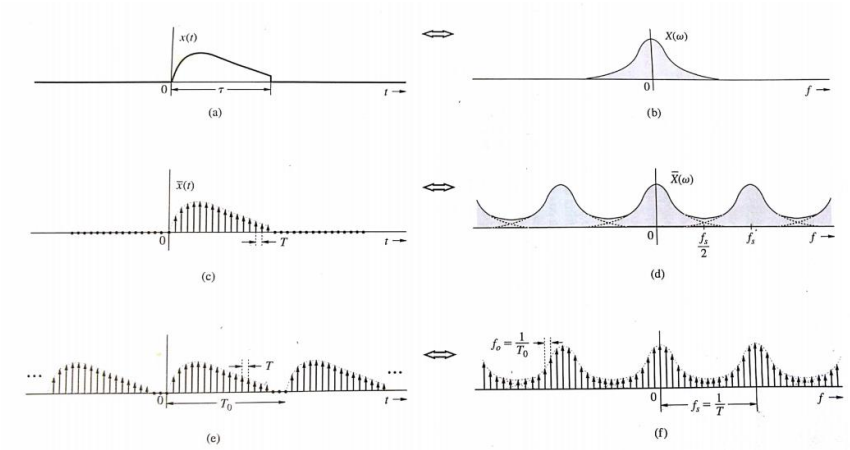
Verifique a utilização do filtro tipo Butterworth na eliminação de ruídos, completando o *script* abaixo.

```
t = 0:0.01:2;
y1= sin(2*pi.*t); % 1 Hz signal
y10 = sin(20*pi.*t); % 10 Hz signal
% soma-se y1 e y10 para gerar um final com ruído em alta frequencia
yt = y1 + 0.3*y10;

%utilizaremos um filtro de sexta ordem, e analisaremos ele até uma
%frequencia de corte menor ou igual a fc = 10Hz. Vemos que o ultimo
%teste elimina totalmente o ruido de alta frequencia (10 Hz)

[b,a] = butter(6,0.4); %distante da frequencia de corte ideal, fc = 10Hz ainda passa
pelo filtro
Filtrado1 = filter(b,a,yt);

[b,a] = butter(6,0.2);
Filtrado2 = filter(b,a,yt);

subplot(5,1,1), plot(t,yt);
subplot(5,1,2), plot(t,Filtrado1); %não funciona
subplot(5,1,3), plot(t,Filtrado2);
subplot(5,1,4), plot(t,Filtrado3);
subplot(5,1,5), plot(t,Filtrado4); %melhor caso
```

Exercício 9

a. Extraído da referência [1] : Um sinal analógico de faixa limitada é amostrado a 7500 Hz (suficiente para assegurar que não haja aliasing), e N amostras são coletadas.
   i. Qual é a resolução em freqüência da DFT em Hz, se N = 1250?
   ii. Para atingir uma resolução em frequência de 4.5 Hz, qual deve ser N?

b. Extraído da referência [2] : Um sinal analógico de faixa limitada é amostrado (N=980, sem aliasing) a 500 Hz. A DFT destas 980 amostras é calculada. Queremos calcular o valor do espectro do sinal amostrado a 120 Hz.
   i. Qual índice *k* da DFT está mais próximo de 120 Hz, e qual é a sua frequência em hertz?
   ii. Qual é o número mínimo de zeros que devemos preencher além das 980 amostras para obter um valor da DFT exatamente a 120 Hz? Qual é o índice *k* da DFT correspondente a 120 Hz?



Relações entre as amostras de $x(t)$ e $X(\omega)$. Figura e discussão abaixo extraídas de [3].

Cálculos numéricos da transformada de Fourier (a DFT) de $x(t)$ necessitam dos valores amostrados de $x(t)$, pois um computador digital pode trabalhar somente com dados discretos. Além disso, um computador pode calcular $X(\omega)$ apenas para valores discretos de ω. Portanto, é umportante relacionar as amostras de $X(\omega)$ com as amostras de $x(t)$. A Figura (a) mostra um sinal limitado no tempo $x(t)$ e, por consequencia, $X(\omega)$ não é limitado em faixa – Figura (b). De aordo com o *teorema da amostragem,* o espectro $\bar{X}(\omega)$ do sinal amostrado $\bar{x}(t)$ é constituído de $X(\omega)$ repetindo a cada f_s Hz, sendo $f_s = 1/T$ como indicado nas Figuras (c,d). No passo seguinte, o sinal amostrado da Figura (c) é repetido periodicamente a cada T_0 segundos, como ilustrado na Figura (e). De acordo com o *teorema de amostragem espectral*, tal operação resulta na amostragem do espectro a uma taxa de T_0 amostras/Hz. Essa taxa de amostragem significa que as amostras são separadas por $f_0 = 1/T_0$ Hz – Figura (f).

A discussão mostra que quando um sinal $x(t)$ é amostrado e, então, periodicamente repetido, o espectro correspondente também é amostrado e periodicamente repetido. Quando amostramos $x(t)$ dentro de um espaço de tempo limitado, $T_0 < \infty$, então, matematicamente, estamos repetindo o sinal periodicamente, pois a transformada de Fourier (para sinal não periódico) é a Série de Fourier (sinal periódico), com período infinito...

Referências

Alguns dos exercícios aqui apresentados foram extraídos e adaptados das seguintes fontes:

[1] Bombois, X. *Signal analyses* http://www.dcsc.tudelft.nl/~xbombois/SR3exercises.pdf
[2] Cuff, P. *Signal analyses* https://www.princeton.edu/~cuff/ele301/files/lecture8_2.pdf
[3] Lathi, B.P. *Sinais e Sistemas Lineares*, 2ª edição, Bookman, 2007.
[4] Oppenheim, A.V. *Signals and Systems*, http://ocw.mit.edu
[5] http://paloalto.unileon.es/ts/first/archives/html/p4_43_0.htm
[6] http://eeweb.poly.edu/iselesni/EL6113/DSP_Exercises.pdf